Curso: Microsserviços na prática com Spring Cloud
Instrutor: Thiago Hernandes de Souza

Prefácio

Curso: Microsserviços na prática com Spring Cloud
Instrutor: Thiago Hernandes de Souza

1 – Introdução

A demanda distribuída de dados tem crescido de forma exponencial nos últimos anos. Tecnologias surgem ao passo que tentam mitigar os famosos "gargalos" em bancos de dados, além de facilitar a comunicação em massa entre APIs.

A **simultaneidade**, a **disponibilidade** e a **escalabilidade** de dados são extremamente importantes nos processos de transformação e entrega de dados.

Sendo assim, o **Apache Kafka** (Figura 1) pode ser uma das soluções de plataforma distribuída de transmissão de dados. Esse é capaz de publicar, subscrever, armazenar e processar fluxos de registros em tempo real. Uma de suas principais características é processar fluxos de dados provenientes de diversas fontes, bem como para entregá-los a vários clientes/fontes.

Figura 1 – Logo Apache Kafka

De uma forma objetiva, o Apache Kafka movimenta volumes imensos de dados não somente de um ponto a outro, mas entre diversos pontos simultaneamente.

2 – Apache Kafka

De acordo com a **Hed Hat** (https://www.hedhat.com), o Kafka foi inicialmente desenvolvido pelo Linkedin para processar **1,4 trilhão de mensagens por dia**, e que agora, se tornou uma solução de transmissão de dados open source aplicável a variadas necessidades corporativas.

Fugindo o contexto do Kafka, ainda em muitas aplicações a integração é realizada por **métodos síncronos**, que utilizam interfaces de programação de aplicações (APIs) para compartilhar dados entre os mais variados tipos de usuários.

Com relação aos **métodos assíncronos**, como segunda opção, e no caso, talvez mais viável, envolve a replicação de dados em um armazenamento intermediário. E é nessa hora que o Kafka entra em cena.

Em suma, o Kafka é uma excelente opção de **integração assíncrona orientada por eventos** que pode aumentar o uso de integração síncrona e APIs, fornecendo mais compatibilidade com microsserviços, além de integração ágil.

O Kafka pode ser aplicado aos mais variados cases de aplicações proporcionando alta produtividade e escalabilidade de dados, além de reduzir integrações point-to-point (P2P). Consegue ainda, reduzir a latência a milésimos de segundos, o que disponibiliza dados aos usuários de forma mais rápida e praticamente em **tempo real**.

Portanto, o Kafka pode ser utilizado em contextos de **Big Data** e **Internet das Coisa** (IoT) facilitando consideravelmente os processos de escalabilidade e proliferação de dados.

3 – Protótipo da Arquitetura

O projeto a ser desenvolvido envolve a arquitetura da Figura 2:

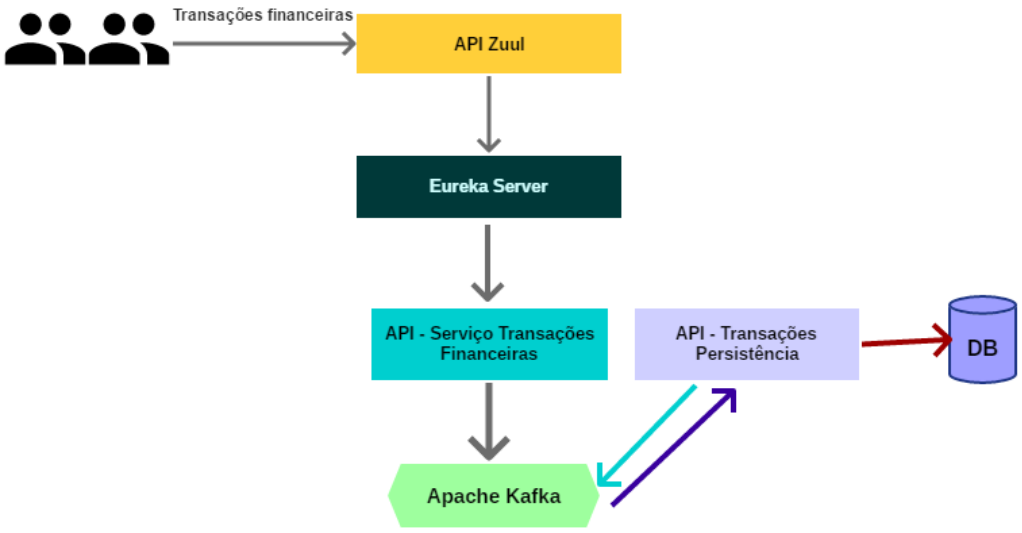

Figura 2 – Protótipo da Arquitetura

4 – APIs – Interfaces de Programação de Aplicações

Envolverá duas outras aplicações: uma voltada para o envio de dados a serem persistidos em banco de dados por meio de uma outra aplicação (de persistência) que receberá esses dados e persistirá em banco de dados.

4.1 – Serviço de Transações Financeiras

Essa API fará o envio de dados para o Apache Kafka, onde esse disponibilizará para a API de persistência de dados que funcionará como um "**listener**" dos dados enviados via métodos POST e inseridos no contexto de tópicos do Apache Kafka.

Os dados serão meros valores de transações financeiras do tipo entrada ou saída.

4.2 – Serviço de Transações Persistência

Será o "listener" e responsável por persistir os dados em banco de dados. Nesse caso, em uma tabela no MySQL.